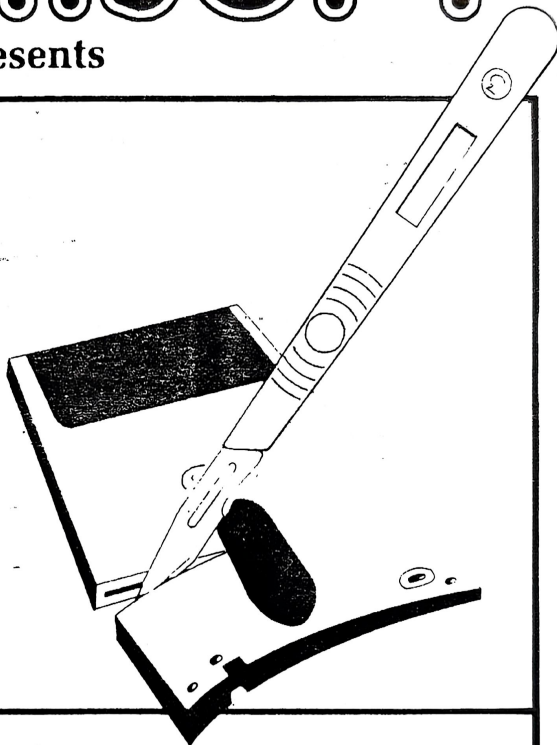


HISOFT

Presents



The Knife

by Aries ♠

Welcome to The Knife

The Knife is a full-feature disc sector editor and file patcher, allowing you to alter individual bytes anywhere on a disc. Unlike other disc editors, though, The Knife comes as two different programs for two different needs. If you want to patch an individual file or if you want to learn about the CP/M disc structure and how files are stored, we recommend you use the first program, called simply Knife. For those of you who want to perform more radical alterations or want to do something quickly, the more powerful but not quite so friendly Knife2 program is likely to be more useful. Although both programs perform approximately the same function, the two are different enough in approach to be explained in separate sections of the manual. The first section is devoted to The Knife and also explains CP/M disc and directory structures and looks at the parts of this structure specific to the Amstrad implementation. There is also a short example to clarify the concepts, which shows you how to *un-erase* a file which you've just deleted. The final part of this manual explains the second program, called Knife2. The tutorial aspects are not covered in such detail in this section, as it is presumed that the user of Knife2 is more familiar with discs. If not, the information contained in part 1 is perfectly applicable to Knife2 as well.

Copyright Notices

The Knife © Copyright Aries Computer Systems 1985

All other programs, documentation and design © Copyright HiSoft 1985,6

All Rights Reserved. No part of this publication may be reproduced or transmitted in any form or by any means, including photocopying and recording, without the written permission of the copyright holder. Such written permission must also be obtained before any part of this publication is stored in a retrieval system of any nature. It is an infringement of the copyright pertaining to The Knife and the associated documentation to copy, by any means whatsoever, any part of The Knife for any reason other than for the purposes of making a security back-up copy of the object code for your own personal use.

Please buy, don't steal.

Backing up your Knife master disc

PCW8256 and PCW8512 owners

We supply the Knife master disc in a format known as *Old System Format*, as this is readable on all Amstrad CP/M computers. Unfortunately, the DISCKIT utility supplied with these machines is unable to copy discs in this format. To make a backup copy, you first must create a new blank disc. Load up CP/M using side 2 of your system disc and type

```
diskit [ENTER]
```

at the A> prompt. After a few seconds the program will be loaded; follow the instructions on the screen to *format* a disc. If you have a second disc drive note that, unless you really want to, there is no point in copying the Knife to a double-sided disc. Once you have formatted a blank disc, exit DISCKIT and replace side 2 of your systems disc in drive A. Now type

```
pip [ENTER]
```

and wait a few seconds. This loads the CP/M PIP utility, which greets you with an asterisk to tell you that it's waiting for input. Remove your systems disc from drive A and replace it with the Knife master disc. Now type

```
m:=*.* [ENTER]
```

and wait until the asterisk returns. You'll notice that PIP produces a message telling you that it is copying, followed by the names of each of the files in turn. When the asterisk returns, remove the Knife master disc from drive A and replace it with the blank disc you have just formatted. Now type

```
a:=m: *.* [ENTER]
```

and again wait until the asterisk returns. This process copies all the files on the Knife master disc to your RAM disc (drive M) and then copies each one back again to your new disc. When PIP's asterisk reappears, hit the [ENTER] key to return to CP/M's A> prompt.

CPC6128 owners

Load up CP/M Plus in the normal way, followed by Amstrad's DISCKIT3 utility program. Follow the instructions to copy a disc, making sure that you copy *from* the Knife master disc to a new, blank disc.

CPC464 with DDI-1 and CPC664 owners

Load CP/M 2.2 and then run the Amstrad-supplied `FORMAT` utility. Follow the instructions to format a new blank disc in *system* format. Now load the Amstrad `DISCCOPY` utility, or `COPYDISC` if you have two drives. Follow the instructions, ensuring that you copy *from* the Knife master disc *to* a new, blank disc.

The Files on Your Disc

If you examine your disc, you should find the following files on it:

```
KNIFE.COM
KNIFE2.COM
KNIFE2J.COM
KNCONFIG.COM
SDIR.COM
WDEL.COM
COPY.COM
PAGE.COM
GENSUB.COM
UNERA.COM
```

There may also be a file called `READ.ME`, as Knife2 and the HiSoft utility programs are updated and improved from time to time; any recent changes will be described in this file which should be listed to the printer and stored with your Knife manual.

Running The Programs

Both The Knife and Knife2 are supplied on the same disc, but Digital Research / Amstrad licensing agreements do not allow us to provide the CP/M operating system as well. This means that it is not possible to *boot* or *warm boot* CP/M from the disc.

This will not affect users of CP/M Plus as provided on the Amstrad CPC6128, PCW8256 and PCW8512 but owners of Amstrad's earlier CP/M 2.2 systems, the CPC464 with DDI-1 and the CPC664 will need to copy CP/M onto their backup copy of the Knife master disc. Use the `SYSGEN` utility, which is provided with the Amstrad machine to put a copy of the system tracks onto the disc so that it is possible to boot from it. Please consult the Amstrad documentation for details of how to do this.

Both The Knife and Knife2 may be run from the CP/M Plus or the CP/M 2.2 operating system. The programs do not mind which version is used and alter themselves accordingly.

The Programs in Detail

The Knife

To start the Knife first make sure that CP/M is running and then insert the master disc into the floppy disc drive and at the CP/M A> prompt type

```
knife [ENTER]
```

where [ENTER] corresponds to a single press of the key marked *ENTER* or *RETURN*. The Knife will load in and display its title screen. Press any key (eg [ENTER]) or just wait a few seconds to move on to the first screen. There is a list of options, often called a *menu*, at the top of the screen:

```
VIEWDISC      FILE:                               EXIT
```

The VIEWDISC option is highlighted in reverse video. To move to the next option press [TAB]. To select the current option press [ENTER].

Viewing a Disc

Select the VIEWDISC option by pressing [ENTER]. The Knife then suggests that the disc will be in drive A. Just press [ENTER] to accept this or type B [ENTER] if you have two drives and want to examine a disc in the second drive.

Now there are some more lines of menu options:

```
DRIVE: A      TRACK: 000    SECTOR: 000
FORWARD BACK SWAP PRINT CHANGE REWRITE
SEARCH
```

and down below are the contents of the first CP/M record. The FORWARD option is highlighted and you can step through each record on the disc by repeatedly pressing [ENTER]. You can go back again by using the BACK option (use [TAB] to step round the options). To examine a particular area of the disc, select the TRACK option and type in the track and CP/M sector (also known as a *record*) number that you require. Tracks go from 0 to 39 and records from 0 to 35, although certain formats of disc created on Amstrad computers may have track numbers from 0 to 159 and record numbers from 0 to 31. If a non-existent track or sector number is entered, The Knife will beep at you.

The SWAP option swaps the contents of the current record with one held in memory. This other record starts off being full of zeros, but by judicious use of the SWAP and REWRITE commands it can be used to transfer data from one record to another. The PRINT option lets you make a printed copy of a record. To alter the contents of a record, select the CHANGE option, make the changes and then select the REWRITE option. Nothing is written to the disc until you select REWRITE, so you can change your mind at any time before this. After you select CHANGE you can type in the new data that you want in hex using [TAB] to move to the next byte. Move down a line with [CURSOR-DOWN] and you can type in character strings. When you finish making changes, press [ENTER] to get back to the menu.

SEARCH lets you find particular patterns on the disc - such as a text string. Select the option and enter the pattern that you are looking for. Before doing so you may enter a *search mask* which allows some of the bits in each byte to be ignored when looking for a match. For example, a search mask of #DF will match regardless of whether letters are in upper case or lower case. Type in the pattern in hexadecimal using [TAB] to enter each byte and type a full-stop after the last byte of the pattern. Alternatively, you may press [CURSOR-DOWN] and then type the pattern as a string of characters - press [CURSOR-DOWN] again to enter some bytes in hex and at the end before typing the full-stop. The Knife will then start searching for your pattern on the disc, going from sector to sector in logical order. You can interrupt the search by pressing any key.

You can return to the top-level menu by using the EXIT option.

Looking at a File

This works in a very similar way to VIEWDISC. Select the FILE option from the top-level menu. An opportunity to select a different user area is given first - this may be defaulted to the normal user 0 by pressing [ENTER]. The records in the file are presented in sequence and you may use the options described above to examine and alter the contents of the file.

If you intend to perform extensive editing of files, we recommend our Devpac80 package which includes ED80 and MON80. ED80 provides comprehensive Wordstar-compatible text editing. MON80 gives extensive capabilities for editing binary program or data files.

Knife2

Knife2 is an alternative disc editor to **The Knife**, offering different facilities and approaching the idea in a completely different way. We think that the two programs complement each other, which is why we supply them both. There are in fact two versions of **Knife2** on the disc we supply; one called **KNIFE2.COM** and the other called **KNIFE2J.COM**. The programs are identical except that the command keys used have been pre-installed for different Amstrad computers. If you have a CPC464, CPC664 or CPC6128, running CP/M 2.2 or CP/M Plus, use **KNIFE2**. If you have a PCW8256 or a PCW8512, use **KNIFE2J**. The rest of the documentation refers to the program always as *Knife2*; users of PCW machines should interpret this as **Knife2j**. As the command keys are different for the two versions, both will be shown in the command descriptions.

Start **Knife2** by typing

```
knife2 [ENTER]
```

at the CP/M prompt. The program clears the screen, prints a sign-on message and then asks you to select a drive. Press A to select drive A or B to select drive B. The program scans the disc in the specified drive, reads in the directory and then displays the first physical sector on the disc. Amstrad disc physical sectors are 512 bytes long in all cases, but **Knife2** shows only 256 bytes at a time. The other half of the sector may be examined at any time by pressing the requisite key. The display looks like this:

```
CP/M Plus DISC SECTOR EDITOR V2.4 by Adam Denning (C) HiSoft 1986
Drive: A   Track: 00   Sector: 01   Block: ****   File: -----

0000 HH HH HH HH HH HH HH HH HH HH HH HH HH HH HH  AAAAAAAAAAAAAAAAAA
0010 HH HH HH HH HH HH HH HH HH HH HH HH HH HH HH  AAAAAAAAAAAAAAAAAA
0020 HH HH HH HH HH HH HH HH HH HH HH HH HH HH HH  AAAAAAAAAAAAAAAAAA
0030 HH HH HH HH HH HH HH HH HH HH HH HH HH HH HH  AAAAAAAAAAAAAAAAAA
0040 HH HH HH HH HH HH HH HH HH HH HH HH HH HH HH  AAAAAAAAAAAAAAAAAA
0050 HH HH HH HH HH HH HH HH HH HH HH HH HH HH HH  AAAAAAAAAAAAAAAAAA
0060 HH HH HH HH HH HH HH HH HH HH HH HH HH HH HH  AAAAAAAAAAAAAAAAAA
0070 HH HH HH HH HH HH HH HH HH HH HH HH HH HH HH  AAAAAAAAAAAAAAAAAA
0080 HH HH HH HH HH HH HH HH HH HH HH HH HH HH HH  AAAAAAAAAAAAAAAAAA
0090 HH HH HH HH HH HH HH HH HH HH HH HH HH HH HH  AAAAAAAAAAAAAAAAAA
00A0 HH HH HH HH HH HH HH HH HH HH HH HH HH HH HH  AAAAAAAAAAAAAAAAAA
00B0 HH HH HH HH HH HH HH HH HH HH HH HH HH HH HH  AAAAAAAAAAAAAAAAAA
00C0 HH HH HH HH HH HH HH HH HH HH HH HH HH HH HH  AAAAAAAAAAAAAAAAAA
00D0 HH HH HH HH HH HH HH HH HH HH HH HH HH HH HH  AAAAAAAAAAAAAAAAAA
00E0 HH HH HH HH HH HH HH HH HH HH HH HH HH HH HH  AAAAAAAAAAAAAAAAAA
00F0 HH HH HH HH HH HH HH HH HH HH HH HH HH HH HH  AAAAAAAAAAAAAAAAAA

First Half
```

The 256 pairs of HH characters above refer to the 2-character hexadecimal representations of the 256 bytes being displayed. The 256 A characters above refer to the ASCII representations of the same bytes. There are 16 bytes to each line and the first four digits on each line represent the hexadecimal offset from the start of the sector of the first byte in the line. The ASCII representations shown are produced by bitwise-ANDing each byte with #7F; if this produces a value in the range #20 (decimal 32, the space character) to #7E (decimal 126, the tilde character) then the character having that value is displayed. Otherwise, a full-stop is printed.

The screen cursor will be flashing over the character representing the high nybble of the first byte displayed. Any keys pressed by you will be interpreted by the program according to whether the key pressed forms a valid hexadecimal character or not; if it does, then its hexadecimal value is placed in the nybble under the cursor and the relevant parts of the display updated accordingly. If the key is a Knife2 command key then that command is invoked. No alterations you make to the sector in memory will be written to disc until you issue either the write sector or paste sector commands. This means that you are free to experiment with data entry and commands until you feel confident. Remember, however, that any alterations you make are your responsibility - playing with discs at this level can easily be disastrous and HiSoft can accept no liability whatsoever for any loss or damage, consequential or otherwise, caused by the use of the programs on this disc.

The cursor may be moved about the display with the four cursor keys. Note that if you have programmed these keys using the CP/M Plus SETKEYS utility, most programs including Knife2 will not be able to respond to them.

The second line of the Knife2 display is known as its *status line* and shows the current drive letter along with the numbers of the track and sector currently being examined. These numbers are in hexadecimal and the track number starts from 00 while the sector number starts from 01. Following the sector number is the CP/M *block* number, also in hexadecimal. If the current sector is not part of a block (i.e. if it falls within the *reserved tracks*) then the block number is shown as ****. The terms *block* and *reserved track* are explained in the section entitled CP/M Disc Structure later in this manual.

The last part of the status line is the filename display. Knife2 works out whether the current sector forms part of a file and if it does, the name of this file is shown. If the sector is not part of a file then the name is shown as -----.

Knife2 Commands

Knife2 responds to a wide variety of commands, many of which are accessed by pressing the [CTRL] key in conjunction with another. This means that the [CTRL] key (marked ALT on the PCW8256 and PCW8512 machines) must be held down while the other key is pressed.

Where the keys used differ between the Knife2 and Knife2j, the Knife2 key is given at the top and the Knife2j key in parentheses at the end of the command description.

Goto next sector

[SHIFT]-Right arrow

This command advances to the next sector on the disc, updating the display as appropriate. If the current sector is the last on a track, then the next sector is the first sector on the next track. If the current sector is the last on the disc then the next sector is the first on the disc (knife2j: use f8/f7 key).

Goto previous sector

[SHIFT]-Left arrow

This command moves back to the previous sector and updates the display. If the current sector is the first on a track then the previous sector is the last on the previous track. If the current sector is the first on the disc then the previous sector is the last sector on the disc (knife2j: use f2/f1 key).

Goto next track

[SHIFT]-Down arrow

This command moves to the next track on the disc and updates the display. If the current track is the last track on the disc then the next track is the first track on the disc (knife2j: use the boxed + key).

Goto previous track

[SHIFT]-Up arrow

This command moves to the previous track on the disc and updates the display. If the current track being edited is the first track on the disc then the previous track is the last track on the disc (knife2j: use the boxed - key).

Show other half of sector

[COPY]

This command updates the display to show the other 256-byte half of the current sector. If the current half is the first half, then the second half is shown. If the current half is the second half then the first half is shown (knife2j: use [EXTRA]-A key).

Hexadecimal/ASCII switch

[TAB]

This command moves the cursor between the hexadecimal and ASCII representations of the sector's contents. When the cursor is over the hexadecimal representations, the data can be altered by typing in new hexadecimal values. When the cursor is over the ASCII representations, the data may be changed by typing in new ASCII characters. Command key values must be entered in hexadecimal, as Knife2 invokes the command if the character itself is typed.

Quit edit

[ESC]

This command quits the current editing session and returns to the prompt asking you to select a drive. Any changes made to the sector in memory but not yet written to disc are lost (knife2j: use [STOP] key).

Search for a string

[CTRL]-F

This command produces a prompt at the bottom of the screen, asking for a string to be searched for. Up to 80 characters may be typed here and the normal CP/M editing keys may be used while it is being entered. To abort the command, press [ENTER] without a string. Once the string has been typed, press [ENTER] to start the search. All searches begin by displaying the current *search mask*, a value which is bitwise-ANDed with the string being searched for and the data being examined before any comparisons are made. This means that a search mask value of #FF (decimal 255) will cause a match to be found only if both strings are identical; a value of #DF will cause a match regardless of the case of any letters in the string. As #DF is often hard to remember, a value of 0 is regarded as being the same as #DF. Type in the new search mask value and press [ENTER], or type [ENTER] by itself to keep the value unchanged.

The search will begin at the start of the next sector and the status line of the Knife2 display is updated to show the track and sector numbers as they are examined. When the specified string is found, the sector in which it occurs is made the current sector and the display is updated to show it on the screen. The cursor is positioned at the first character of the string. If the string cannot be found by the time the last sector has been examined, the program prints an error message and, upon receiving a keypress, returns to the track and sector at which the search was started.

The search may be aborted at any time by pressing a key. When you do this, the sector which was under examination at the time becomes the current sector and the display is updated accordingly. Note that for a string to be found it must occur wholly within a physical sector, so if a string cannot be found on first search, it often is worthwhile trying again with a shorter portion of the same string (knife2j: use [EXTRA]-F key).

Search for a sequence of bytes

[CTRL]-G

This command is very similar to search for a string above except that the prompt asks for a sequence of bytes, each of which may be entered in hexadecimal or decimal. End the sequence by pressing [ENTER] alone to the Byte: prompt. The command may be aborted if it is mistakenly invoked by pressing [ENTER] alone in response to the first Byte: prompt. Once the sequence has been typed the search proceeds in the same way as the string search; you will be asked for a new search mask value and the search then begins. Note that for a sequence to be found it must occur wholly within a physical sector, so if a sequence cannot be found on first search, it often is worthwhile trying again with a shorter portion of the same sequence.

Search for next occurrence

[CTRL]-N

This command will search for the next occurrence of the last-entered string or byte-sequence. The search begins at the sector following the current sector and the command allows a new search-mask value to be entered.

Write sector

[CTRL]-W

This command causes the current sector contents in memory to be written back to disc at the location of the current sector. Any changes you have made will be made to the disc, too, if you issue this command, so ensure that the data are correct before you do so!

Paste sector

[CTRL]-Q

This command causes the current sector contents in memory to be written back to disc at a location you specify; it asks you for the new track and sector numbers and then writes the sector to that location. Any changes you have made will be made to the disc, too, if you issue this command, so ensure that the data are correct before you do so!

Select new sector

[CTRL]-S

This command prompts for a new sector number, which may be entered in decimal or in hexadecimal, and then makes that sector the current sector. If the sector number you give does not exist, you are re-prompted. The command may be aborted by pressing [ENTER] without entering a new number.

Select new track

[CTRL]-T

This command prompts for a new track number, which may be entered in decimal or in hexadecimal, and then makes the current sector number on the new track the current sector. If the track number you give does not exist, you are re-prompted. The command may be aborted by pressing [ENTER] without entering a new number.

Select new block

[CTRL]-B

This command prompts for a new block number, which may be entered in decimal or in hexadecimal, and then makes the first sector of the new block the current sector. If the block number you give does not exist, you are re-prompted. The command may be aborted by pressing [ENTER] without entering a new number.

Go to start of named file

[CTRL]-D

This command prompts you for a filename which should be entered in the normal way with up to 8 characters of primary filename and then, if there is an extension, the delimiting full-stop followed by up to 3 characters of extension. The letters may be entered in upper or lower case, or a mixture of both.

The directory is searched for this filename and if it is found in any user area then the program calculates the first block number holding this file and makes the first sector of that block the current sector. The display is updated accordingly. If the file cannot be found then an error message is displayed and you are re-prompted.

Filenames may be entered in short-form and Knife2 will attempt a simple pattern-match. The short-form is simply as many characters of the name as you feel like entering. This part-name is searched for in the directory and is matched by the first filename which matches in the characters you have entered. For example, if your disc contains the files

```
knife2.com  
knife2j.com  
knife2.gen
```

in that order in the directory, a part-name of knife2 will find knife2.com, one of knife2j will find knife2j.com and knife2.g will match knife2.gen. This does have the slightly disadvantageous side-effect of making knife2.c match KNIFE2.COM rather than KNIFE2.C if the former occurs first in the directory. To enter filenames with short extensions like this so that they will match with the correct files, pad out the extension the requisite number of spaces on the right.

Go to next block in extent

>

When tracking a file across a disc the ability to go to the next block in the file with a single keypress is extremely useful. This command allows you to do so within the confines of one extent (actually one directory entry but most Amstrad discs have only one extent per directory entry). If the current sector is not part of a block, is part of a block which is not occupied by a file or is part of the last block in an entry, the command is ignored. Otherwise the first sector of the next block is made the current sector and the display is updated.

Go to previous block in extent

<

This command allows you to go to the previous block in a file within the confines of one extent (actually one directory entry but most Amstrad discs have only one extent per directory entry). If the current sector is not part of a block, is part of a block which is not occupied by a file or is part of the first block in an entry, the command is ignored. Otherwise the first sector of the previous block is made the current sector and the display is updated.

How Knife2 knows the format of a disc

When you start an editing session of Knife2, it needs to know various details about the disc to be edited. It finds this out by asking the CPM BIOS (Basic Input / Output System) for the address of one of its data structures known as the *Disc Parameter Block* (DPB). The DPB holds vital facts such as the number of sectors a disc has, how big a CPM block is, how many directory entries there are and so on. Knife2 extracts the data it needs and uses it in many of the calculations it needs to perform to work out block numbers, filenames and so forth.

The BIOS used in Amstrad machines contains even more information than this, but Knife2 doesn't use it directly. This allows the knowledgeable user to patch the DPB for a particular drive and make Knife2 read a disc which has a strange format or which was not created on an Amstrad. For example, it is a very simple matter to alter the DPB to allow Knife2 to read PCW8256 discs on a CPC6128.

Details of how a DPB may be patched are not given here as it is no simple matter. If you are interested in this aspect of Knife2, you would need to be thoroughly familiar with CPM, the BDOS and BIOS and the format of the discs you want to examine. Official Digital Research documentation is probably the best source of most of this information.

Regardless of the physical numbering of sectors, Knife2 always displays sector numbers as beginning from 01.

Configuring Knife2

To enable you to alter the keys used to invoke commands in Knife2 and Knife2j, we have provided a special program called KNCONFIG.COM which prompts you for new values and then alters Knife2. The program is run by typing its name at the CPM prompt. As it *resets the disc system* there must be a disc in drive A while the program is running. The content of that disc is not important.

The program contains instructions for use which are displayed when it is run. It is fairly straightforward and enables you to alter the key values in a version of Knife2 on any disc and under any name. The configured version is saved under the same name as the original, overwriting the contents. The Knife2j program was created from Knife2 in this way.

Configured versions of Knife2 may be reconfigured at any time in the same manner.

The CP/M Disc Structure

Now that you have your copy of *The Knife*, you can begin to hack away at discs to your heart's content. However, for the program to be really useful a full understanding of the way CP/M organises files on the disc is most important. Every disc formatted on a CP/M system has a similar structure, and the Amstrad computers' disc system is no exception. Once certain basic disc *parameters* have been discovered, the organisation of the disc can be relied upon to be thoroughly predictable. Before we look at how we can determine and use these parameters, we must understand the concepts involved in floppy disc technology as a whole. This means we must know what is meant by *sectors* and *tracks*.

Sectors and tracks

A floppy disc is essentially a piece of plastic covered in a magnetic film which is very similar to that found on cassette and video tapes. A tape system has the disadvantage that, in order to read or write a particular piece of data, the length of tape preceding the required area has to be traversed by the read/write head before the data can be acquired. Disc drives get over this problem by moving the head as well as the magnetic film. The disc itself is spinning at a very high speed and the head is able to move along a radius of the disc, covering any given annulus at any given time. Obviously, a motor is required to move the head across the disc's surface.

We can control a motor extremely precisely if we allow it to move in pre-defined steps, so it follows that if each step corresponds to a part of the disc which contains (or is to contain) data, the data can be recorded and retrieved with great speed and considerable accuracy. To facilitate this process, the data is stored in a given number of circular 'tracks' on the disc. The number of tracks per disc is fixed at the time of manufacture; the two most common standards are to have 40 tracks or 80 tracks. In the case of a 40-track disc, such as used by most of the Amstrad disc formats, this means that the entire working area of the disc is divided into forty annular regions, to each of which the read/write head may be moved.

The amount of data which may be held in a track is often very substantial and as a result most machines are unable to handle it all in one go. To solve this problem each track is divided up into divisions called *sectors*. The number of sectors within a track is a function of the system's hardware and software and is one of the disc parameters we mentioned earlier. Likewise, the amount of data which may be held in a sector is determined by another parameter. This is known as the *sector size*; common sector sizes are 128, 256, 512 and 1024 bytes - the Amstrad discs always have 512-byte sectors.

In most cases the Amstrad's discs are formatted to have 9 sectors of 512 bytes each per track and as there are 40 tracks it follows that a side of a disc may hold up to $40 * (9 * 512)$ bytes which works out as 184320 bytes, or 180K. When a disc is formatted, it sets up the sectors and marks the sectors within a track with a number. The first sector in a track is usually numbered 0 or 1, but Amstrad has chosen to use strange values such as 65 (#41) and 193 (#C1) for some of its disc formats.

With our newly-gained information on tracks and sectors, we can see that it is possible to refer to a given sector's data by its track number and by its sector number. Track numbers always start at 0, so the very first sector on a normal Amstrad disc is

Track 0 Sector 01

and the last sector on such a disc is

Track 39 Sector 09

The discs known as CF2-DD, which go into drive B on the PCW machines, have 160 tracks, numbered from 0 to 159.

The ability to refer uniquely to a sector is suprisingly useful, as we are about to find out, but first a warning:

The CP/M operating system believes that sectors always are 128 bytes long. These mythical sectors are also known as *records*, *logical sectors* or *CP/M sectors*. Throughout the rest of this guide we call them *records*. When the Knife program displays a sector number it is talking about a record. The real *physical sectors* on the disc are 512 bytes long and we refer to them here as *sectors*. The Knife program makes no reference to physical sectors. The Knife2 program, however, does exactly the opposite and refers always to physical 512-byte sectors rather than CP/M's logical sectors.

CP/M Discs

The CP/M operating system deals with data in groups of 128 bytes regardless of the physical configuration of the sectors. The 128-byte groups are known as *records* in CP/M terminology and happen to be the smallest amount of data which the operating system acts on with reference to discs. This means that every CP/M disc is an exact multiple of 128 bytes in length. Now read again the large print warning above! The next step up in data handling is the *block*, which is a group of records belonging to the same file. A block is usually one of 1024, 2048 or 4096 bytes long, which means that there will be either 8, 16 or 32 records per block.

All Amstrad disc systems except the PCW CF2-DD discs have eight records (= 1024 bytes) per block, which means that a block occupies two physical sectors. The sectors will be contiguous in numbering terms, so if the first block occupies sectors 01 and 02 of track 2 the second block will occupy sectors 03 and 04 of the same track. Files always occupy a whole number of blocks, even if the physical length of a file would seem to dictate otherwise. This means that a file which is five records long will occupy exactly the same amount of disc space as one which is seven records long. It is clear that the smaller the block size, the more efficient the use of disc space, but at the same time a small block size reduces the speed with which a file may be read or written. A compromise is arrived at and as floppy discs are now rather cheap, a large block size is often adopted. The directory of a CP/M disc always occupies block 0 and possibly other consecutive blocks as well. The directory is the place in which the system keeps a record of all the files on the disc and where they can be found. Once we've found the first directory block, the rest is easy. But how do we calculate the location of the directory? This is where the disc parameters are required.

The information we need is shown below for each of the Amstrad formats. Remember that not all Amstrad machines support all the disc formats so if there is one you do not recognise, don't worry!

Amstrad IBM Format

Single-sided

512-byte physical sector size

40 tracks numbered 0 to 39

1024-byte block size

64 directory entries

8 sectors per track numbered #01 to #08

1 reserved track

Amstrad System / Vendor Format

Single-sided

512-byte physical sector size

40 tracks numbered 0 to 39

1024-byte block size

64 directory entries

9 sectors per track numbered #41 to #49

2 reserved tracks

Amstrad Data Format

Single-sided

512-byte physical sector size

40 tracks numbered 0 to 39

1024-byte block size

64 directory entries

9 sectors per track numbered #C1 to #C9

0 reserved tracks

Amstrad Joyce Format (normal PCW8256 format)

Single-sided

512-byte physical sector size

40 tracks numbered 0 to 39

1024-byte block size

64 directory entries

9 sectors per track numbered #01 to #09

1 reserved track

Amstrad CF2-DD format (PCW8256 / PCW8512 drive B format)

Double-sided

512-byte physical sector size

160 tracks numbered 0 to 159

nnn directory entries

9 sectors per track numbered #01 to #09

1 reserved track

Some of this information is not new to us, but the part which gets us to the directory is the number of *reserved tracks*. A CP/M disc almost always has some reserved tracks. These tracks may be used to hold the code required to boot the system into the machine and also may house the CPM *BDOS* (Basic Disc Operating System) and the *CCP* (Console Command Processor). Reserved tracks always start from the beginning of the disc so if the format says '2 reserved tracks' it means that tracks 0 and 1 are used by the system for its own purposes.

The directory follows the reserved tracks, so the first directory record of a disc with 2 reserved tracks must be

Track 2 Record 0

Let's examine the first few bytes of a typical directory to find out how these things are laid out:

```
0000 00 50 49 50 20 20 20 20 20 43 4F 4D 00 00 00 3A .PIP COM...:
0010 02 03 04 05 06 07 08 09 00 00 00 00 00 00 00 00 .....
0020 E5 43 4F 55 4E 54 20 20 20 47 45 4E 00 00 00 14 .COUNT GEN...:
0030 0A 0B 0C 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
```

It's obvious from this that each directory entry occupies 32 bytes, which enables us to gain yet more information about the disc structure. If we take this entry as being part of an Amstrad system format disc you'll remember that the disc parameters mentioned that there are 64 directory entries. If each directory entry is 32 bytes long, the entire amount of disc space devoted to the directory must be $32 * 64$ bytes, which is 2048 bytes or 2 blocks. This means in turn that the directory on a system format disc occupies sectors 0-7 on track 2. So now we know where the directory starts and how long it is. As it occupies two blocks on an Amstrad system format disc, it follows that the first block free for data is block 2. Let's take a look at an individual directory entry and see what information we can gain from this

```
0000 00 50 49 50 20 20 20 20 20 43 4F 4D 00 00 00 3A .PIP COM...:
0010 02 03 04 05 06 07 08 09 00 00 00 00 00 00 00 00 .....
```

The very first byte of the entry, which is '00' here, tells us the status of the file. If this byte is 0, as above, it means that the file exists. If it is #E5, the file has been deleted. A deleted file's directory entry can be overwritten by the next file to be created on the disc and all files whose first directory byte is #E5 will not be found in a CP/M BDOS *open* call. As well as 0 or #E5, the first byte may also be any number from 1 to 15. This means that the file exists, but in a different *user area*. Normally when using CP/M we are in user area 0 but we can issue the built-in USER command to move to another user area. The main use of this is on large (hard) discs where directories can get too large to comprehend unless they are split into different areas. You can also hide files by putting them into a different user area.

The next eight bytes of the entry comprise the file name. The characters are always stored by the system as upper case and the field is right-filled with spaces (CHR\$(32)). The three bytes following this comprise the file extension, which is again in upper case and right-filled with spaces. If you make any alterations to the directory entry, remember to use only valid characters in the filename and extension fields and use only upper case letters as otherwise you may not be able to access the file!

The full-stop which we use to separate filenames and extensions, as in

GEN80 MYFILE.GEN

is not stored in the directory; there is no need as the name and extension fields are always eight and three bytes long respectively.

To simplify the next part of the discussion, we're going to ignore something called an *extent* until a little later. The sixteenth byte of a directory entry, which is 3A in our example above, tells us how many records there are in the file. We can see that PIP.COM has 3A records, which is 58 in decimal. As there are 8 records per block, it follows that PIP.COM will occupy 8 blocks, but the first two records only in the final block form part of the file. So far so good, but how do we know where to start looking on the disc for PIP.COM? The answer lies in the next few bytes. These bytes, from byte 17 in a directory entry onwards, are the numbers of the blocks occupied by the file, in the order in which they were written. All Amstrad disc formats except the PCW CF2-DD discs use block numbers between #00 and #FF, which means that the numbers fit into a byte. These are referred to as 8-bit block numbers. The CF2-DD disc format has more than #FF blocks, so the numbers have to be stored in 16 bits. Hence, they are referred to as 16-bit block numbers.

We can see that PIP.COM occupies blocks 2, 3, 4, 5, 6, 7, 8 and 9. We also know that all 8 records in blocks 2 to 8 are part of PIP, but only the first 2 records in block 9. Block 0 starts at track 2 record 0, block 1 at track 2 record 8, and block 2 at track 2 record 16, which is the first record containing part of PIP. It follows through sequentially like this, so it's dead easy to find every single record of a file. There are times when a file will not occupy consecutive blocks as other, smaller, files may have been deleted and the space re-used by the new file. Nevertheless, it is not at all difficult to translate block numbers from a directory entry into physical sectors on a disc, and thus locate every record of a file. Knife2 even does this for you. There is one problem, though...

Extents

As the last 16 bytes of a directory entry are used to hold the block numbers occupied by the file, it follows that each directory entry may reference only 16 blocks, which in the case of the Amstrad means 16K. This is a bit of a limitation as we are sure to want to read and write files which are longer than 16K from time to time. CP/M gets over this by introducing the concept of an *extent*, each of which occupies one directory entry and can therefore be no longer than 16K. If a file is too long to be held in one extent, the CP/M BDOS opens another extent for it, creating a directory entry identical to the previous one except that the new entry reflects only the number of records in this extent and the block numbers refer to this extent only. If the file needs yet another extent, the process continues, and so on.

Each directory entry, although referring to a different part of the file, has the same filename entry, so how do we tell which directory entry refers to which extent? You guessed it! One of the spare directory entry bytes is used to tell us which extent this entry refers to. The byte in question is byte 13, which is the first byte following the last character of the filename extension. The first extent of a file has this byte set to 0, the second to 1 and so on. For the same reason that individual parts of a file may not necessarily be saved on contiguous blocks, so extents may not necessarily appear in order in the directory. It all depends on where the free space is. If, when you examine a file's directory entry on an Amstrad CP/M or Amsdos disc, you find that it doesn't refer to the first extent, keep on looking. Likewise, when you read an entry and find that there are 128 (#80) records in that file, check to see if there are any more extents recorded in the directory. There may not be, as there is nothing to stop a file being an exact multiple of 16K bytes long, but you must check to make sure.

A problem with extents is that high-capacity discs with large block sizes may store more than more than one extent in each directory entry. This is shown by having a field in the *DPB* (Disc Parameter Block) set to the number of extents per directory entry minus 1. This field is known as *EXM* (the extent mask) and it usually is therefore 0. If the *EXM* value is non-zero, there is more than one extent per entry, and the byte in a directory entry referring to the number of records occupied is valid only for the highest-numbered extent in this entry; all lower-numbered extents must by definition have #80 records.

The final pieces of information held in a directory entry are cleverly hidden away in the 3-byte filename extension field. These bytes are normally capital letters or other ASCII characters, which means they are below #80 (128) in value. This in turn means that the very top bit, bit 7, is zero in all cases. As CP/M knows this, it takes advantage of the fact and uses this bit to store vital pieces of information. The top bit of the first character of the extension tells the system whether the file is set to *R/W* (read/write) or *R/O* (read-only) access. If the bit is clear (zero), as it is normally, the file is considered to be a read/write file, while if it is set (1), the file is read-only. The top bit of the second character of the extension is used to signify whether the file is a *directory* (DIR) file or a *system* (SYS) file. A '1' corresponds to SYS and a '0' to DIR. A file marked as SYS will not be listed in normal directory listings.

The HiSoft Utility Programs

The programs supplied with The Knife and Knife2 are useful CP/M utilities designed to fulfil a range of purposes. Some will be useful to all users of discs and CP/M, others may be useful only to programmers. The important things about these programs from HiSoft's point of view are that they are small and easy-to-use. Each is described below; note that as each one is written to execute under CP/M, the files are stored on the disc as the name given plus the extension *.COM*.

An extremely powerful version of Knife2, called **Knife86**, is available for PC-DOS and MS-DOS machines which are IBM PC-compatible. This program can edit floppy and hard discs as well as non-DOS discs and knows all about DOS 2 features such as sub-directories and clusters. Please call HiSoft for further details.

WDEL

This program allows you selectively to delete files on a disc. WDEL is invoked by typing its name followed by a standard CP/M *ambiguous file specification* such as `B : *.COM`. If no ambiguous file specification is given, WDEL treats it as if `*.*` has been entered. If a drive specification only is entered, such as `B :`, WDEL treats it as `*.*` on the specified drive.

The name of each file which is found to match the file specification is then displayed along with a prompt

```
Delete (Y/N/A/Q)?
```

If you hit the `Y` or `y` key, the displayed file is deleted. If you press the `A` or `a` key, this and all subsequent files which match the specification are deleted without further prompting. If you press `Q` or `q`, the program ends and returns to CP/M without deleting the current or subsequent files. If you press any other key, the file is not deleted and the name of the next matching file is displayed.

If any file which is set to Read Only or is protected in some other way is selected for deletion with WDEL, the program prints a message

```
Cannot delete!
```

and continues to the next file.

SDIR

This utility produces and displays a more detailed listing of the files on a disc than that produced by the built-in DIR command. Like DIR, SDIR may be invoked by itself or it may be followed by a CP/M ambiguous file specification such as A:KNIFE.*. For example, to get a complete listing, type

```
sdir [ENTER]
```

or

```
sdir a: [ENTER]
```

and so on. To get a listing of all files with the .COM extension, type

```
sdir *.com [ENTER]
```

or

```
sdir b:*.com [ENTER]
```

etc.

The listing produced on the screen is made to fit into 80 columns and consists of up to two filenames per screen line. Each name is shown in full, followed by an S if the file's *system* bit is set and followed by an R if the file's *read-only* bit is set. A file may have both attribute bits set, in which case both S and R are shown. Please see earlier sections of this manual for a complete description of system and read-only attribute bits.

These one-letter flags are followed by the length of the file in K (1024-byte units), rounded up to the nearest 1K. This number is displayed in decimal. This number is followed by the letter R to make its meaning clear. After this comes the number of *records* in the file. A record is a 128-byte unit used by CP/M as the basis of all its disc operations. A file always occupies an integral number of records whatever its physical length. The number displayed is in decimal. When all the filenames which match the ambiguous file specification have been displayed, SDIR prints the amount of disc space free in 1K units.

SDIR has the advantage over CP/M utilities such as STAT, SHOW and DIR [FULL] in that it is less than 1K long; the other utilities are between 6K and 15K long. SDIR does not sort the filenames into alphabetical order.

COPY

COPY is a short-and-sweet file-copying utility which makes the task of transferring given files between discs relatively easy. It is used like this:

```
copy <afn> {<destination drive name>} {-q} [ENTER]
```

(The curly brackets, { and }, specify an optional item)

where <afn> is a standard CP/M *ambiguous file specification* from a lone drive name such as A: up to a complete filename such as A:COPY.COM. Being an ambiguous file specification, the standard CP/M wildcard characters, ? and *, may be used to match a group of files. Note that a lone drive name is taken as an ambiguous file specification which matches all files on the specified disc in the current user area. The <destination drive name>, if present, is a normal CP/M drive name such as B:. If left out, the default drive is used. The -q is the *quiet* option switch and its action will be described shortly.

COPY finds each file matching the <afn> given and presents the name of each one to the user along with a prompt:

```
Copy (Y/N/A/Q) ?
```

Y corresponds to Yes, N to No, A to this and **all** subsequent matching files and Q to Quit. Each file to be copied is copied to the destination drive with the same name as it has on the source drive. Any file of the same name is deleted prior to the operation whether the copy succeeds or not, unless it specifically has been marked *read-only*.

If the -q option is present (the q may be in upper or lower case) in the command line then the Copy (Y/N/A/Q) ? prompt will not appear. COPY will run as if A had been pressed in response to this prompt before any file names were displayed. In other words all files matching the <afn> are copied without prompting. Note that the -q option must be after the <afn> and the <destination drive name> (if present) in the command line.

If the source and destination drives are the same an error message is printed and no names are produced. If the copy fails for lack of disc or directory space a message is printed to the console and the new copy of the file is deleted.

PAGE

PAGE is an extremely useful program which prints out a number of files to the program's *standard output*. It is usual to re-direct this output to the printer or to a file to be spooled at a later date. Output may be redirected in the standard Unix / MS-DOS manner by specifying a filename preceded by the > character in the command line. The named file (or device) is opened and all PAGE's standard output is sent to the file rather than to the screen. When PAGE terminates, the file is closed. Note that sign-on messages and file status reports are sent by PAGE to its *standard error*, which is not re-directable and appears always on the screen. The filename used to redirect standard output (which also is the screen until told otherwise) may be a CP/M device such as LST: (the printer) and PUN: (the serial port).

The most useful features of PAGE are that each of the important paper characteristics may be altered and that command lines may include wildcards (ambiguous filenames) to specify a group or groups of files.

The general form of a PAGE command line is

```
page <afn> {<afn> <afn> etc} {<options>} [ENTER]
```

If page by itself is typed at the CP/M prompt the usage message is produced. This is a very brief summary of the command-line syntax and the options available. Each option may be placed anywhere in the command line and each one affects all files. An option is introduced with the - character, which is followed immediately by the single-letter option. Four of the option letters take a decimal number as a parameter and this must be placed immediately following the option letter. The options available are:

B	Set bottom margin
F	Don't use formfeeds
L	Set line length
N	No output; show matching filenames only
P	Set page length
Q	Quiet; don't output status messages
S	Single-sheet printing; pause after each page
T	Set top margin

The N option is useful when you want to see what files you have specified before paging them. The S option is useful when sending output to a sheet-feed printer; the program pauses after each page has been output and waits for a key to be pressed before proceeding to the next page.

A typical PAGE command line would be:

```
page -p50 *.c myfile.asm -f print.me! >lst: -s [ENTER]
```

This causes all files with the extension .C to be paged along with MYFILE.ASM and PRINT.ME!; the output is sent to the printer (>lst:) and the page-length is set to fifty lines. The requisite number of linefeeds, rather than formfeeds, are used to separate pages and the program pauses after each page.

PAGE sends each file to the standard output in pages, with the name of the current file printed in the top left-hand corner and the current page number in the top right-hand corner. The filename is over-printed by sending it to the printer followed by a lone carriage-return and then sending the name again. On almost all printers, this causes the name to be over-printed, giving an emboldened effect.

The default page-length is 66 lines, the line length is 80 characters and the top and bottom margins are 3 lines each. Formfeeds are used to separate pages of output but this may be altered with the -F option.

GENSUB

This program is used to generate *batch* (submit) files. It generates a list of filenames matching a given or defaulted CP/M ambiguous file specification (*afn*) and precedes each one with a given command. Optionally a list of parameters may be appended to each line, and GENSUB is able to recognise certain special characters and perform specific actions on meeting these. The general form of a GENSUB command line is:

```
gensub [-e] {<afn> {<first command> {<command tail>} } } [ENTER]
```

where items enclosed in curly brackets (braces) are optional. The `-e` option, if present, must be the first item on the command line. All items in the command line have default values but if an item in the list is included, all preceding items above also must be included. The `-e` option is an exception to this rule - it may be left out if not required.

Each line of output produced by GENSUB is sent to its *standard output*. This is usually the screen but may be re-directed in the standard Unix / MS-DOS manner by specifying the file or device name to receive the output preceded by a `>` character in the command line. See the description of PAGE above for more details on redirection.

The output produced by GENSUB is formed as follows:

```
First, the item specified by <first command> is output
Then, the current filename is output
Finally, the items specified by <command tail> are output
```

If `<command tail>` is not present, it defaults to no items; if `<first command>` is not present, it defaults to COPY. The `<afn>` parameter, if not specified, defaults to `*.*`. This parameter may consist only a drive name, such as `M:`, in which case it matches all files on the specified drive in the current user area.

The `<command tail>` item, as it is the last item on GENSUB's command line, can be an arbitrarily-long list of items. `<first command>`, however, is determined by its position in the command line and thus can comprise only one item. To get around this limitation, `<first command>` may be a list of items enclosed wholly in double quotes:

```
"command1 -aflag another theend"
```

GENSUB offers a limited amount of macro expansion. If the dollar sign, `$`, is found anywhere in `<first command>` or `<command tail>` it is treated specially by the program. The dollar sign itself is discarded and the character immediately following the dollar sign is interpreted in the following way:

@	is expanded to the full filename of the current file
*	is expanded to the name only (no extension) of the file
_	is expanded to a newline sequence
\$	is used to represent the dollar sign itself

So if GENSUB was invoked with this command line

```
gensub *.c "cc -n" $_as -zap $*_ln $* -lmylib -ls -lc >doit.sub
```

and the disc contained the following .C files

```
first.c
myprog.c
gensub.c
```

then when GENSUB finished the file DOIT.SUB would contain the following lines:

```
CC -N FIRST.C
AS -ZAP FIRST
LN FIRST -LMYLIB -LS -LC
CC -N MYPROG.C
AS -ZAP MYPROG
LN MYPROG -LMYLIB -LS -LC
CC -N GENSUB.C
AS -ZAP GENSUB
LN GENSUB -LMYLIB -LS -LC
```

which could then be executed using the Digital Research SUBMIT utility. Neither <first command> nor <command tail> may exceed 1023 characters after expansion, although no checks are made by the program to see if this is adhered to. As SUBMIT itself is able to handle lines no longer than 127 characters, this should not be a problem.

UNERA

UNERA is an extremely useful utility which can *unerase* a file which has just been removed with the ERA or WDEL commands. It can work only if no other alterations have been made to the disc or directory. In other words, if you have edited the disc with The Knife or Knife2, if you have subsequently saved another file or altered the directory in any way, UNERA will probably *not* work!

It is most useful immediately after the injudicious use of ERA or WDEL. Simply type

```
unera <filename> [ENTER]
```

where <filename> is the name of the file you want un-deleted. The program will try its best to recover the file but please remember the restrictions and warnings given above. Good luck!

Other Programs from HiSoft

Many other language and utility programs are available for your Amstrad computer from HiSoft. Among these are Devpac80 (macro assembler, front-panel debugger, full-screen editor), Pascal80 (standard Pascal compiler, very fast), C++ (Kernighan & Ritchie integer C compiler), FTL Modula-2 (superb, full Modula-2 compiler with interactive editor), Write-Hand-Man (like Borland's Sidekick), TurboBASIC (integer BASIC compiler for the CPC series), Font64 (character designer / screen dumps on your CPC machine), Art in Pascal (CPC graphic art program *complete with the source in Pascal libraries*), The Torch (an interactive CP/M tutorial) and Catalog (organise your discs into a database).

Full details of all these programs and the latest developments are available from:

HiSoft

The Old School Greenfield
BEDFORD MK45 5DE
Tel: (0525) 718181

Acknowledgements

The Knife was written by Aries Computing Ltd

Unera was written by HiSoft

All other programs were written by Adam Denning of HiSoft, using the Gen80 Z80 assembler, 8086 assembler, HiSoft C and Aztec C.

